

**AMENDMENTS TO THE CLAIMS:**

This listing of claims will replace all prior versions and listings of claims in the application:

1. (Previously Presented) A method for validating programs, the method comprising:
  - receiving a meta-language description of a computer program, the meta-language description comprising a definition module and an implementation module, the implementation module defining a first class to be implemented by the program and the definition module defining a first interface associated with the class;
  - validating the meta-language description;
  - generating a language-dependent program from the meta-language description, the language-dependent program comprising the first interface and the first class; and
  - performing usage and semantic checks by compiling the generated first interface and the generated first class.
2. (Previously Presented) The method of claim 1 wherein validating the meta-language description comprises validating the syntax of the definition module and the implementation module.

3. (Cancelled).

4. (Cancelled).

5. (Withdrawn) A method for validating programs, the method comprising:

- receiving a language-independent description of a computer program, the language-independent description comprising a definition module and an implementation module;
- validating the language-independent description;
- generating a language-dependent program from the language-independent description, the language-dependent program comprising a script code section in a language that does not support interfaces; and
- validating the language-dependent program.

6. (Withdrawn) The method of claim 5 wherein validating the language-dependent program comprises:

- extracting language elements from the script code section; and
- comparing the extracted language elements with the definition module.

7. (Withdrawn) The method of claim 6 wherein extracting language elements comprises generating a symbol table from the script code section.

8. (Withdrawn) The method of claim 5 wherein generating the language-dependent program comprises:

generating language-dependent code comprising an interface and a class.

9. (Withdrawn) The method of claim 5, wherein validating the language-dependent program comprises:

extracting language elements from the script code section;

comparing the extracted language elements with the definition module;

generating language-dependent code comprising an interface and a class;

and

compiling the interface and the class.

10. (Withdrawn) A method for validating programs, the method comprising:

receiving a language-independent description of a computer program, the language-independent description comprising a definition module and an implementation module;

validating the language-independent description;

generating a first language-dependent program from the language-independent description, the first language-dependent program comprising a first script code section in a language that does not support interfaces;

generating a second language-dependent program from the language-independent description, the second language-dependent program comprising a second script code section of a distinct, second kind in a language that does not support interfaces;

extracting a first set of language elements from the first script code section;

extracting a second set of language elements from the second script code section; and

comparing the first set of language elements and the second set of language elements with the definition module.

11. (Previously Presented) A computer program product, tangibly embodied in a computer-readable storage device, the computer program product comprising instructions operable to cause data processing equipment to:

receive a meta-language description of a computer program, the meta-language description comprising a definition module and an implementation module, the implementation module defining a first class to be implemented by the program and the definition module defining a first interface associated with the class;

validate the meta-language description;

generate a language-dependent program from the meta-language description, the language-dependent program comprising the first interface and the first class; and

perform usage and semantic checks by compiling the generated first interface and the generated first class.

12. (Previously Presented) The computer program product of claim 11, wherein the instructions to validate the meta-language description cause the data processing equipment to validate the syntax of the definition module and the implementation module.

13. (Cancelled).

14. (Cancelled).

15. (Withdrawn) A computer program product, tangibly embodied in a computer-readable storage device, the computer program product comprising instructions operable to cause data processing equipment to:

receive a language-independent description of a computer program, the language-independent description comprising a definition module and an implementation module;

validate the language-independent description;

generate a language-dependent program from the language-independent description, the language-dependent program comprising a script code section in a language that does not support interfaces; and

validate the language-dependent program.

16. (Withdrawn) The computer program product of claim 15, wherein the instructions to validate the language-dependent program cause the data processing equipment to:

extract language elements from the script code section; and  
compare the extracted language elements with the definition module.

17. (Withdrawn) The computer program product of claim 16 wherein the instructions to extract the language elements cause the data processing equipment to generate a symbol table from the script code section.

18. (Withdrawn) The computer program product of claim 15, wherein the instructions to generate the language-dependent program cause the data processing equipment to:

generate language-dependent code comprising an interface and a class.

19. (Withdrawn) The computer program product of claim 15 wherein the instructions to validate the language-dependent program cause the data processing equipment to:

extract language elements from the script code section;  
compare the extracted language elements with the definition module;

generate language-dependent code comprising an interface and a class;  
and  
compile the interface and the class.

20. (Withdrawn) A computer program product, tangibly embodied in a computer-readable storage device, the computer program product comprising instructions operable to cause data processing equipment to:

receive a language-independent description of a computer program, the language-independent description comprising a definition module and an implementation module;

validate the language-independent description;

generate a first language-dependent program from the language-independent description, the first language-dependent program comprising a first script code section in a language that does not support interfaces;

generate a second language-dependent program from the language-independent, the second language-dependent program comprising a second script code section of a distinct, second kind in a language that does not support interfaces;

extract a first set of language elements from the first script code section;

extract a second set of language elements from the second script code section; and

compare the first set of language elements and the second set of language elements with the definition module.

21. (Previously Presented) An apparatus, comprising:

means for receiving a meta-language description of a computer program, the meta-language description comprising a definition module and an implementation module, the implementation module defining a first class to be implemented by the program and the definition module defining a first interface associated with the class;

means for validating the meta-language description;

means for generating a language-dependent program from the meta-language description, the language-dependent program comprising the first interface and the first class; and

means for performing usage and semantic checks by compiling the generated first interface and the generated first class.

22. (Previously Presented) The method according to claim 1, wherein the language-dependent program comprises a script code section written in a scripting language.

23. (Currently Amended) The method according to claim ~~25~~ 22, further comprising:

generating a compiler-language representation of the script code section, the compiler-language representation of the script code section comprising a second interface and a second class; and

performing a semantics check of the script code section by compiling the second interface and the second class.

24. (Currently Amended) The method according to claim 23 22, further comprising:

generating a compiler-language representation of the script code section, the compiler-language representation of the script code section comprising a second interface and a second class; and

performing usage and semantic checks by compiling the generated second interface and the generated second class.

25. (Previously Presented) The method according to claim 22, further comprising performing a usage check on the script code section by:

extracting language elements from the script code section; and

comparing the extracted language elements with the meta-language definition module.